This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

Claim 1 (Previously Presented): A method of scheduling CPU resources comprising the steps of:

using a counter to determine when to allocate the CPU resources;

instructing an interrupt controller, via non-maskable interrupts from the counter, to allocate the CPU resources; and

instructing the CPU to allocate resources in real-time by the interrupt controller issuing non-maskable interrupts to the CPU.

Claim 2 (Original): The method of claim 1 wherein only a portion of the CPU resources are allocated.

Claim 3 (Original): The method of claim 1 wherein all of the CPU resources are allocated.

Claim 4 (Original): The method of claim 2 wherein the CPU resources are allocated to at least one thread, and the CPU resources are allocated by determining a duration of time and a periodicity for execution of said at least one thread.

Claim 5 (Original): The method of claim 3 wherein the CPU resources are allocated to at least one thread, and the CPU resources are allocated by determining a duration of time and a periodicity for execution of said at least one thread.

Claim 6 (Original): The method of claim 1 wherein the counter is a performance counter.

Claim 7 (Original): The method of claim 6 wherein the performance counter counts machine cycles in order to determine when to allocate the CPU resources.

Claim 8 (Original): The method of claim 6 wherein the performance counter counts executed computer instructions.

Claim 9 (Cancelled).

Claim 10 (Cancelled).

Claim 11 (Cancelled).

Claim 12 (Cancelled).

Claim 13 (Previously Presented): A method of scheduling resources on at least one microprocessor that includes a CPU and a device, the method comprising the steps of:

using the device to determine, in response to a first non-maskable interrupt, when to allocate the resources in real-time;

causing the device to issue a second non-maskable interrupt to the CPU when it is time to allocate the resources; and

causing the CPU to allocate the resources in response to the second non-maskable interrupt.

Claim 14 (Original): The method of claim 13 wherein the device is a performance counter.

Claim 15 (Original): The method of claim 13 wherein the device is a timer.

Claim 16 (Previously Presented): A method of scheduling resources on at least one microprocessor that includes at least one performance counter, at least one programmable interrupt controller and at least one CPU, said method comprising the steps of:

allowing the CPU to execute a first thread;

using the performance counter to determine when to allocate the resources to a second thread on a real-time basis;

issuing a first non-maskable interrupt from the performance counter to the programmable interrupt controller when it is time to allocate the resources to the second thread;

instructing the programmable interrupt controller to issue a second non-maskable interrupt to the CPU that instructs the CPU to switch execution from the first thread to the second thread;

instructing the CPU to stop execution of the first thread;

causing the CPU to store first current state information regarding execution of the first thread;

causing the CPU to restore second current state information regarding execution of the second thread; and

allocating resources to the second thread.

Claim 17 (Original): The method of claim 16 wherein the programmable interrupt controller is an APIC.

Claim 18 (Original): The method of claim 17 wherein the microprocessor is selected from the group consisting of: a Pentium 4GB, a Pentium Pro 64GB, a Pentium MMX 4GB MMX, a Pentium II 4GB MMX, a Pentium III 4GB MMX KNI, a Celeron 4GB MMX, a Xeon PII 64GB MMX and a Xeon PIII 64GB MMX KNI.

Claim 19 (Previously Presented): A computer-readable medium having computer-executable instructions stored for performing steps comprising:

using a scheduler to control execution of at least one thread based on a second non-maskable interrupt issued by an interrupt controller;

using at least one counter to issue a first non-maskable interrupt to the interrupt controller to notify the interrupt controller to issue the second non-maskable interrupt to notify the scheduler when to switch execution of said at least one thread on a real-time basis.

Claim 20 (Cancelled).

Claim 21 (Previously Presented): The computer-readable medium of claim 19 wherein said at least one counter is a performance counter and counts CPU cycles.

Claim 22 (Previously Presented): The computer-readable medium of claim 19 wherein said at least one counter is a part of a CPU and counts executed instructions.

Claim 23 (Previously Presented): The computer-readable medium of claim 19 further comprising instructions for executing said at least one thread at a highest IRQ level.

Claim 24 (Previously Presented): The computer-readable medium of claim 19 further comprising instructions for executing said at least one thread in a transparent manner so that at least one operating-system process is unaware of the execution of said at least one thread.

Claim 25 (Original): The computer-readable medium of claim 24 further comprising instructions for executing all of said operating-system processes and all of said at least one threads as a single real-time thread.

Claim 26 (Cancelled).

Claim 27 (Original): The computer-readable medium of claim 19 further comprising instructions for allocating at least a portion of a CPU's resources to an operating-system process and using the remaining CPU resources for execution of said at least one thread.

Claim 28 (Original): The computer-readable medium of claim 27 further comprising instructions for releasing the CPU resources back to the operating-system process when said at least one thread finishes execution.

Claim 29 (Original): The computer-readable medium of claim 27 further comprising instructions for releasing the CPU resources to another thread when said at least one thread finishes execution.

Claim 30 (Original): The computer-readable medium of claim 19 further comprising instructions for allocating a predetermined number of CPU cycles for execution of an operating-system process and using the remaining CPU cycles for execution of said at least one thread.

Claim 31 (Cancelled).

Claim 32 (Cancelled).

Claim 33 (Cancelled).

Claim 34 (New): A computer-readable medium having computer-executable instructions for performing real-time execution-thread switching comprising:

issuing a first non-maskable interrupt from a counter to an interrupt controller when the counter turns over;

in response to receiving the first non-maskable interrupt, issuing a second non-maskable interrupt from the interrupt controller to a central processing unit;

in an interrupt service routine that services the second non-maskable interrupt,

saving a first execution thread's current state information,

setting the counter to specify when the counter will turn over again,

restoring previously stored state information pertaining to a second execution thread; and

after execution of the interrupt service routine has finished, executing the second execution thread.

Claim 35 (New): The computer-readable medium of claim 34, wherein the counter is an advanced programmable interrupt controller.

Claim 36 (New): The computer-readable medium of claim 34, wherein the first execution thread's current state information includes stack data, processor data, and floating point-unit data.

Claim 37 (New): The computer-readable medium of claim 34, wherein the previously stored state information pertaining to a second execution thread includes stack data, processor data, and floating point-unit data.

Claim 38 (New): The computer-readable medium of claim 34, wherein the second execution thread is executed after interrupts, which were pending when the interrupt service routine finished, have been executed and after deferred procedure calls, which were pending when the interrupt service routine finished executing, have been executed